A Data Distribution Strategy for the 90s (Files Are Not Enough)

Mike Tankenson
Steven Wright
Telos Systems Group
Jet Prepulsion laboratory

Virtually all of the data distribution strategies being contemplated for the EOSDIS era revolve around the usc of files. Most, if not all, mass storage technologies rrrc based around the file model. However, files may be the wrong primary abstraction for supporting scientific users in the 1990s and beyond. Other abstractions more closely matching the respective scientific discipline of the end user may be more appropriate. JPL has built a unique multimission data distribution system based on a strategy of *telemetry stream emulation 10* match the responsibilities of spacecraft team arid ground data system operators supporting our nations suite of planetary probes.

The current system, operational since 1989 and the launch of the Magellan spacecraft, is supporting over 200 users at 15 remote silts. This stream-oricn[cd data distribution model can provide important lessons learned to builders of future data systems.

JPL's Multimission Ground Data System (MGDS)

JPL's MGDS is a distributed, workstation based, ground data system that provides or~-line. near-line and off-line storage for all telemetry, ancillary and processed data in support of the Voyager, Magellan, Galileo, Mars Observer, and Ulysses missions. in the future the MGDS will support the MESUR Pathfinder mission. the CASSINI mission to Saturn, and the mission to Pluto currently in the early planning stages. The MGDS began development in 1985 as the Space Flight Operations Center (SFOC) software upgrade following the successful prototyping effort to apply workstation technology to support the Voyager encounter with Uranus and continues through today as part of the Advanced Multimission Operations System (AMMOS) with mission support and maintenance activity.

The MGDS provides a Project Data Base (PDB) for each mission Consisting of two parts:

- A Telemetry Record-Based System.
- A File-Based System to support data products processed at levels 2 and above,

The file-based system is in c1osc harmony with systems proposed for EOS. The tile storage system consists of science and engineering file data products, and a catalog constructed using relational database technology (Sybase). The MGDS supplies a variety of tools for browsing the catalog and importing and exporting products to and from the system.

The tclcmctry-record based system, the subject of this paper, consists of the set of all Level 0 and selected Level 1 mission telemetry products and related ground data system information. Specifically, the tclcmctry-records based system contains:

- Spacecraft Engineering Data
- Decommutated (channelized) Spacecraft Engineering Data
- Level O and Level 1 Science Data
- Deep Space Network (DSN) Monitor Data
- Radio Science Data
- Quality, Quantity and Continuity (QQC) Data.

The telemetry-record base system is implemented as the Telemetry Delivery Subsystem (TDS) and supported by the Central Database Subsystem (CDB).

The MGDS System Architecture is based around a set of project Local Area Networks (LANs) interconnected over a high speed backbone (Figure 1). Wide Area LANs are supported to the Magellan spacecraft team in Denver, and to PIs/CoPIs all over the country for Mars Observer. Each project LAN has a CDB for non-real-time data storage, and a TDS for near real-lime (NERT) and real-thnc telemetry data access. The basic architectures of these two systems are common among projects -- typically, project specific adaptations require chances to tables along with minimal software changes.

The Spacecraft Team and Operations Support

The Telemetry Delivery Subsystem's primary role is to support the daily activities of the individual Mission Spacecraft Teams, the supporting Multimission Control Team (MCT) and Data System Operations Team (11 S03'), and to provide science investigators with access to their primary data. The function of the Spacecraft Team is to operate the spacecraft, monitor its health, perform routine calibration and maintenance and deal with spacecraft anomalies. Spacecraft teams consist of spacecraft subsystem analysts (power, propulsion, command and control, . . .), a navigation team, telecommunications analysts and others typified by the Mars Observer team with over 40 personnel (including management and staff). During periods of routine operation, the Spacecraft Teams at JPL operate on a 40-hour S-day work week asa baseline. On a typical work clay, MGDS users will review engineering and ground data system data received since the end of the previous working day, and continue with real-time data throughout the day. Each element within the Spacecraft Team will summon data related to their area of responsibility from the TDS for processing and analysis.

The MCI' and DSOT provide 24-hour monitoring of all spacecraft and operation and control of the JPL ground data system. For these teams, the primary role of the TDS is to provide operators with data to support problem resolution.

Query Requirements

Typical scenarios supported by the Telemetry Delivery System include daily queries from the end of the previous working day right into the current real-time stream of return link telemetry; a 10-to 60-day trend analysis study; a query for retransmitted data from the DSN; ad hoc queries of on-line engineering data to support anomaly resolution on the spacecraft or ground data system and a query for science data from a local or remote principle investigator. To support all of these responsibilities the *data distribution strategy of emulating telemetry streams* was devised. A telemetry stream consists of a subset of processed telemetry data tailored to the needs and responsibilities of the user.

The strategy to emulate telemetry streams allows TDS to support on-line, interactive access to telemetry, access to real-time return link streams, and to provide seamless queries that transition from noa-real-tin] ctorcal-tin]ctclcmctrydata, Gap filling, overlap removal, besting are supported automatically and transparently. Because the data distribution model is based on a functional rather than an implementation model of the system, users can interact with the data system based on their operational view of the system with little or no knowledge about file systems, database manager internals, or data [transmission protocols.

Telemetry Record System Organization

To support oar distribution strategy the telemetry data system is organized by mission, telemetry record type and, more fundamentally, by time. There is a plethora of clocks within the scheme of mission telemetry to support. Spacecraft Clock (SCLK), Spacecraft Event Time (SCET), Earth Receive Time (ERT), Record Creation Time (1{C'I'), Monitor Sample Time (MS]'), Radio Science Sample Time (1{SS'1'), and even orbit number was proposed as a clock. Each of these clocks have unique behaviors which affect

the ordering of data. SCLKs arc subject to spacecraft reset and tape recorders anomalies such as the "crap-in-the-gap" phenomena where old data was recorded and remained in between new recording periods. This old data is streamed back imbedded in the latest recorded data. SCETs are corrected for reset, but will be effected by "crap-in-the-gap". ERT is a simple, well-behaved clock, but is not homomorphic with the spacecraft clock because of recorder playback.

The clocks of primary interest to the spacecraft teams arc SCET, and ERT. The preference typically reflects whether a mission has a short one-way light lime (Magellan) where team members tend to work in terms of SCET, or a long one-way light time (Voyager) where team members seem to prefer ERT. The data system must handle both on an equal footing and produce a stream of telemetry that is ordered "as it occurred". Thus, queries by ERT arc ordered by SCET unless specifically requested otherwise. To support this level of functionality, the telemetry database had to go through several incarnations,

The Magellan Telemetry Database

Magellan was the pathfinder system and our first attempt to implement the telemetry stream access strategy. The approach taken on the Magellan system was to implement a *Channel database* (Borgen, [3]) in addition to a telemetry record database. To understand the Channel database we need some background. Planetary spacecraft (and presumably earth orbiting spacecraft) usc the concept of commutation and decommutation to pack and unpack telemetry data during transmission to Earth. Commutation occurs on the spacecraft, and involves systematically sampling several sources of data and constructing a single telemetry frame from the.sc samples. Each sample occupies an assigned position (as specified in a decommutation map) in a regular, repeating fashion. Decommutation occurs on the ground, where separation of the single telemetry frame into its component parts takes place based on their assigned position (as specified in the same decommution map) in a data frame. The *channelization* process is performed on the data after acquisition by matching each sample value with an explicit channel identifier. I'bus, a channel is the output data from a single instrument or sensor, uniquely identified by the MGDS.

In the JPL telemetry world, there arc various types of channels. Engineering channels correspond directly to spacecraft instruments and sensors. Monitor channels arc added to the telemetry stream by the Deep Space Network (DSN) where tracking data, radio science and other quality indicators arc produced. QQC channels arc added to the telemetry stream by the Product Generation System and represent the processing analysis done on the raw data. Header channels arc those values that correspond to tile SFDU CHDO headers (discussed below) that arc added to the data as pari of telemetry processing. All of these channel data arc packaged in the same manner anti archived for later distribution to users.

This Channel database was an experiment where "channelized" telemetry was disassembled, anti the individual channel records were stored into a relational database. The premise of the system was that users would be able to perform complex operations on channels using a relational model, and that the performance would be superior. In term of performance, the Channel database was fairly fast for queries, but the loading suffered duc to the overhead of loading thousands of data items into an RDBMS (compared to loading a file of data). There was also the problem that channels arc dynamic and can be added to the system at any time by changing the commutation process on the spacecraft, or by introducing new channels through the DSN. Thus the PDB had to be able to dcai with both time and channels and dynamic variables. This early experimental system was not pretty, but was able to formulate a tailored stream of telemetry in response to a request by a user. This capability formed the basis of our fundamental strategy.

MGDS Multimission Telemetry Database Architecture

After the Magellan system went operational (and it is still in operation), the telemetry database was redesigned to enhance its multimission nature and resolve the issues associated with the Channel

database. The redesign took advantage of the VANESSA prototyping effort to support Voyager's Neptune Encounter. VANESSA placed greater emphasis on storing and retrieving telemetry records rather than individual channels in satisfying the needs of the science community for near real-ti[nc (NERT) access to data. Any channels needed were extracted "oa-[lie-tly", either by the Query Server or by the users analysis tools. The channel database was dropped in the redesig n and a simpler storage mechanism was implemented for near real-time data base.d on files ([he NERT cache). To maintain our distribution strategy in the new system, data is separated by telemetry record type as it is recorded into files. These files arc cataloged according to record type, and the start and end clocks of interest for that type. Data is queried from the NERT cache through a process of ordering the tiles according to their starting clocks and time merging the data across them.

The most challenging complication in this approach has been dealing with clock anomalies. In order to support queries by any of the clocks mentioned above, and to be able to order the data by any of those clocks "on-the-fly", the clocks associated with the data within any file have to be well behaved. This means that for all clocks of interest, the end time has to be greater than the start time, and clock values have to be monotonically increasing. To guarantee this behavior, algorithms were devised to detect clock anomalies as the data is being loaded. When anomalous behavior is detected, loading to the current file of data is closed out (and cataloged) and a new file started. If the new file has well-behaved clocks (just disjoint from the previous file), loading contin ues. If the clocks are poorly behaved they are isolated and query processing may or may not ignore them base.ci on the query request.

The VANESSA prototype also had the capability to provide real-time access to data as it entered the system. Users of the VANESSA prototype were able to query from the past and into the future and receive stored and real-timcda[a *in the same query.* This capability had been specified for the original Magellan system but was never implem ented. Although the PDB provide near real-tim e loading of telemetry data, access to future data was impossible to implement in the context of an RDBMS because these systems will only support queries of data already existing within the database. The simpler NERT cache storage model has made it possible to implement a real-time query capability and provide data to end users directly as it is received and processed from the DSN.

Finally, the initial concept of the NERT cache was as a *short term storage location.* Responsibility for longer term on-line storage was retained by the CDB subsystem. The NERT cache was intended to provide quick access to data and smooth out the operational irregularities so loading data into the CDB. The NERT cache has proven to be very robust and use of the CDB telemetry record storage system is starting to wane. Nevertheless, the final query system as implemen ted in the TDS provides seamless access to all three sources of data (CDB, NERT cache and real-time). Its architecture is illustrated in Figure 2.

Application of Standards, The Standard Formatted Data Unit

The Standard Formatted Data Unit (SFDU) has been critical to the development of the MGDS and its data storage system. SFDUs provide a way to globally define and identify data products for interchange among various software applications a nd international organization s (Miller, Elgin, [2]). The SFDU concept provides a means for globally defining and identifying data products; a means for aggregating instances of these data products; and a means for administering the data products definitions and descriptions to ensure their accessib ility and understanding. The abstract nature of the SFDU has proven itself time and time again in constructing software to meet JPL's multimission requirements by providing sufficient polymorphic richness to characterize all telemetry data within the system.

The SFDU structure is derived from Label-Value-Objects (LVO) which arc self-idcn(ifyinp, and self-clclimi[il]gda[a records that follow the labeling rules of the Consultative Committee on Space Data Systems (CCSDS) or one of its Control Authorities. LVOs have a label element to identify the data object and give it length, and an element that contains the data values (data fields). High level SFDU structure guidelines are determined by the CCSDS and focus on standard labeling of data. These guidelines include

rules to enable individual agencies to define their own detailed formatting specifications. JPL has adopted or developed several standards for formatting data within the SFDU including the Compressed Header Data Object and the Parameter Value Language (PVL).

Compressed Header Data Objects

A Compressed Header Data Object (CHDO) is an LVO except that it has a shortened, 4-byte label to provide a compact envelope structure for telemetry, monitor and QQC data. The CHDO structure is used only for data exchange between MGDS subsystems. The CHDO label contains a 2-byte type field and a 2-byte length field. The fixed size of the length field places a 32-kilobyte limit on the size of CHDO-structured SFDUs. The type field contains an integer representation of type information sufficient for MGDS purposes (Figure 3).

CHDOs at JPL are enveloped within SFDUs with standard CCSDS labels, making the SFDU readable by other systems that use the SFDU standard. Within the SFDU header itself, JPL further defines subheaders (Figure 4):

- Aggregation subheader CHDO
- Primary subheader CHDO (required: data type, mission ID)
- Secondary subheader CHDO (optional: mission independent metadata)
- Tertiary subheader CHDO (optional: mission dependent metadata)
- Quaternary subheader CHDO (optional: mission dependent metadata)
- Data CHDO

The data ("metadata") fields of the primary, secondary, tertiary, and quaternary subheader CHDOs further define and identify the data, The headers are produced by the MGDS Product Generation System, and may be mission independent or mission specific. The content of these subheaders are defined by the projects.

Parameter Value Language

The Parameter Value Language (PVL) is a simple ASCII language of the form "keyword = value;" plus some delimiting constructs. PVL provided a standard for expressing query requests, in ASCII, that could be encapsulated within an SFDU in a standard fashion (Figure 5, TDS Query Protocol).

Data Aggregation

The Version 3 SFDU label provides the ability to create a variable length information product without requiring byte counts of the product's length. This was utilized by TDS to create an SFDU compliant query product that could be constructed and transmitted to the end user, on-the-fly, without having to stage the product locally to measure its size and fill in the label of' the encapsulating SFDU. The Version 3 SFDU labels support (in addition to others) the notion of delimiting an SFDU by an End Marker, The marker is embedded in the length field of the encapsulating SFDU and is paired with an End Marker Label at the end of the data product (Figure 6, TDS Data Product).

Stream-based Versus File-based Data Distribution

"Get away from files and filenames" (Dozier, [4])

The easy way to manage data distribution problems involving extremely large datasets is to use files. The file model is universally understood and supported by all operating systems, storage systems and network transfer services. In addition, once the requested data is staged into files, there is nothing more for the data system to do other than to notify users to retrieve them. Presumably, users will have access to plentiful file transfer tools (commercial or public domain) and can perform the actual transfer

themselves. Once the files rrrc transferred, the job of the data system is complete -- it is the user's problem to get at the scientific data within the files.

XBROWSE, from the University of Rhode Island **(URI)**

In contrast, data systems based on streams or other abstractions require more processing and system administration support, but enhance the usefulness of the system 10 end users. The 'xbrowse' system pr-ovidcs one such example.

The 'xbrowse' system, developed at URI (Kowallski, Gallagher, et al [1]) is a stream-based layer over a data system whose basic data abstraction presented (o the end user is the image. Users make requests for images that, because of their size, arc broken up into 'chunks' by sampling the high resolution images and transmitting a stream image of progressively higher resolution. The data is transmitted directly into data visualization tools at the client site (which 1M% be local to URI, or over the Internet). The system allows users 10 view images and to throttle the incoming data interactively if the image is examined at low resolution and rejected. A file-based system would not bc able to provide either of these capabilities directly.

Telemetry Output Tool

Users of JPL's MGDS arc provided with an interactive, point and click (and type a little bit) telemetry query tool called the Telemetry Output Tool (TOT). Users arc presented with an abstraction that closely models the Telemetry problem domain. Figure 7 shows the TOT graphical user interface with widgets for selecting packets, channels, channel sets, lime ranges, spacecraft, clock types, and so on. Once users have specified the query parameters for TOT (including the desired output), transfers occur 'in the background'. The requested data is packaged into standard SFDU objects and, if requested, delivered directly into workstation analysis tools [such as the MGDS Data Monitor & Display, (DMD)] over local and wide area networks. Users interact with the system via the telemetry stream abstraction with no knowledge of the underlying file or database management systems involved.

The 'look and feel' of the TOT interface is the same for all JPL missions. Each mission "adapts" the TOT through MOTIF resource files (TOT is constructed using the public domain Widget Creation Library, WCL, which affords considerable flexibility) rather than constructing new query applications for each new mission because the underlying abstraction is derived from the model for doing business at JPL.

Building Custom Client Tools

As mentioned above, abstract views of a data system require extra processing by the system. Both '1'01' and 'Xbrowse' required custom software, at the client side, 10 properly present the system and ingest their data products. Unlike the file transfer model where standard FTAM and FTP tools can bc assumed, no standards exist to construct these client tools. A first step in developing standard data system presentations in clicn[-side software is to adopt some existing standards for data packaging (SFDU, HDF, etc.), and then provide enhanced client/server tools that understand the formals. To some extent, the NCSA tools supporting HDF arc built on this model.

Although neither XBrowse nor TDS provide a general solution 10 representing wcc data systems to users, both arc good examples of developing presentations to data system users which more closely model their particular problem domain.

References

1. J.G. Kowalski, J.H. R. Gallagher, A. Reza Nekovei, P.C. Cornillon. Remote Access to Multi-Inventory Oceanographic Digital Data Archives. See also: J. Gallagher, P.C. Cornillon. "AVHRR Imagery and In-Situ Data Accessed Via Internet", EOS: Transactions: American Geophysical Union, Volume 74, No. 17, April 2.7, 1993, p. 204.

2. D. Miller and B. Elgin. introduction to the Advanced Multimission Operations System (AMMOS) Lecture Course. December 18, 1992. M6 MOPS0511 -00-05.

3. R. Borgen. "The Unique Problems of Using Relational Databases for Space Missions", DBA ShopTalk, Database Programming and Design, December 1991,

4. J. Dozier. Presentation the at the August 1992 EOSDIS Quarterly Review, Greenbelt Ml).

Acknowledgments

This work was done under government contract to the Jet Propulsion Laboratory and the California Institute of Technology.

Figure 1: MGDS Six-Mission Configuration

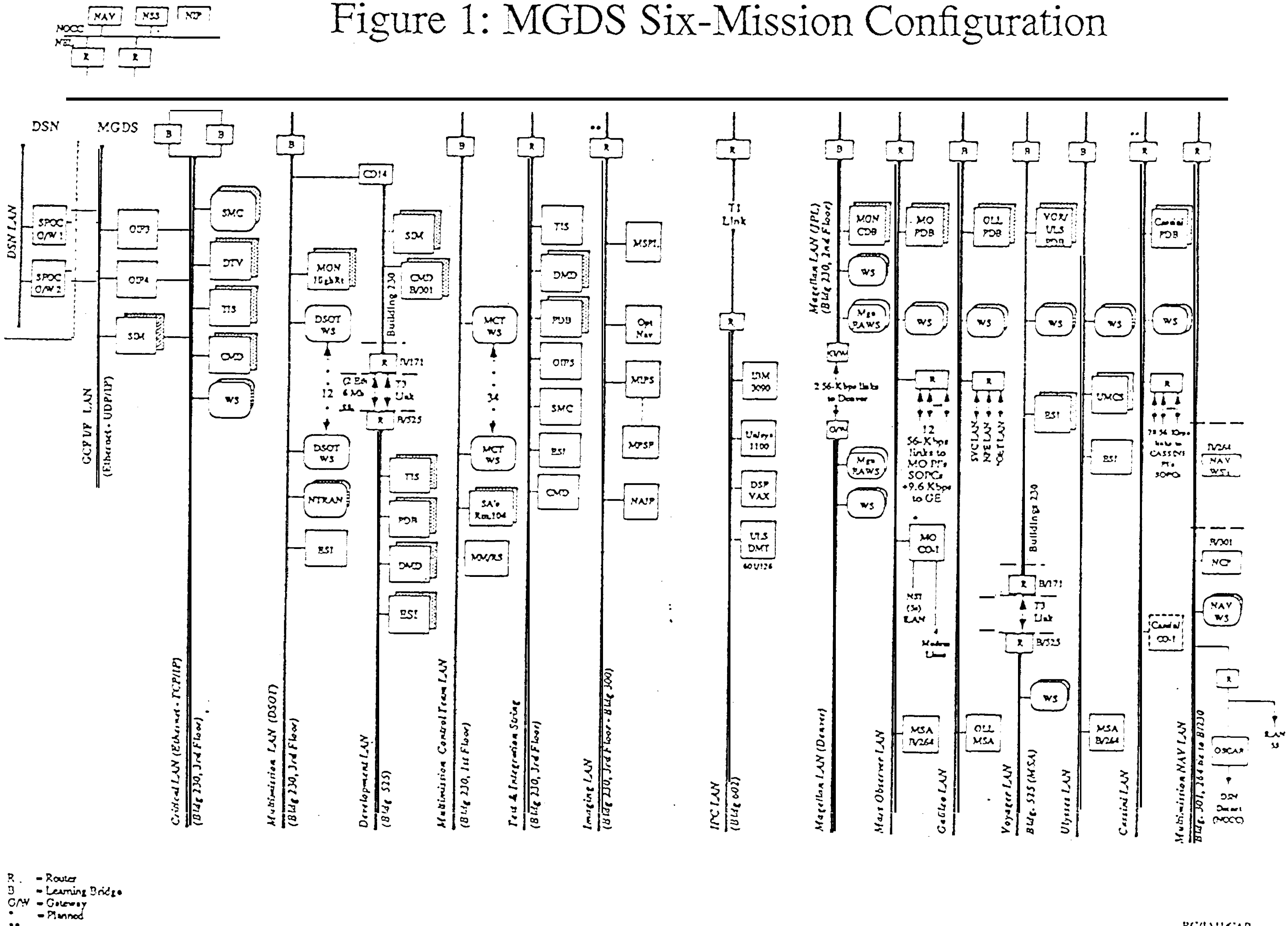

Figure 2: TDS Server Architecture

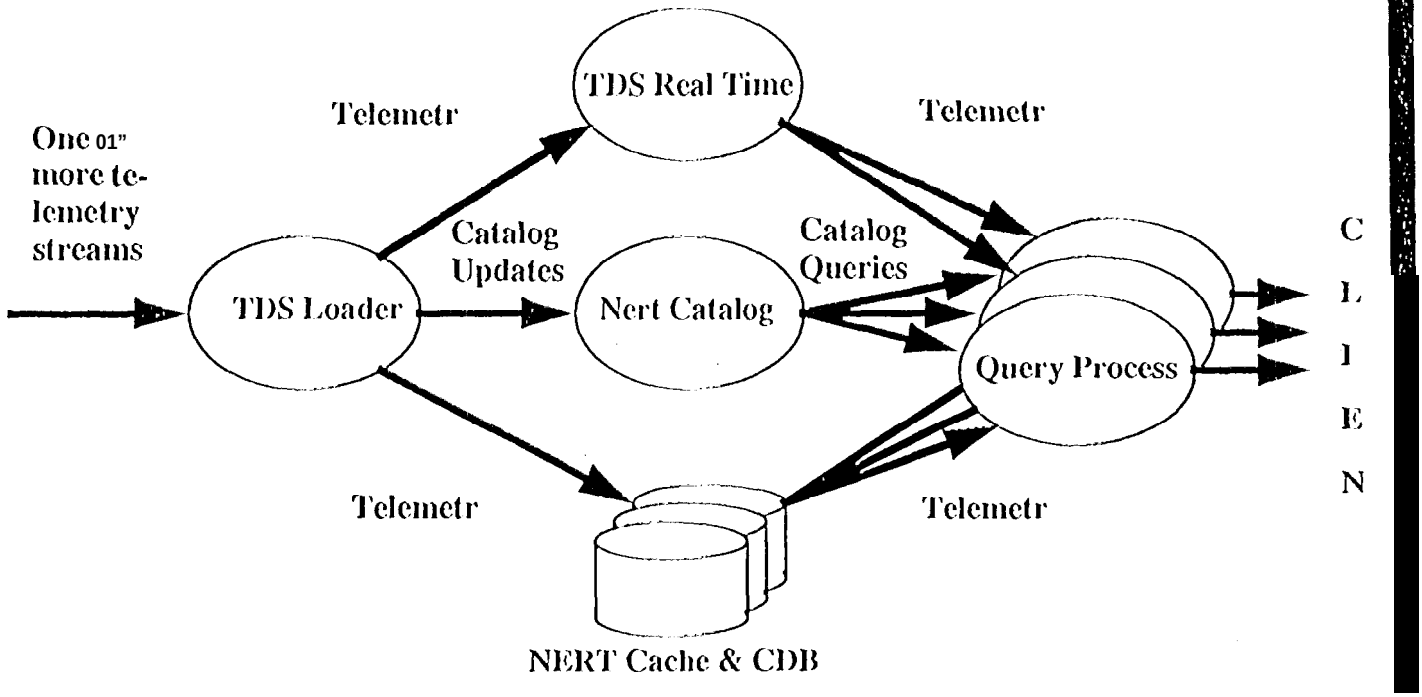

Figure 3: Compressed Header Data Object

Type	Control Authority ID, Version ID, Class ID, Spares, 1)1)11)	Fixed Field Size
Length	Length of Value Field	Fixed Field Size
Value	Data /Information or Data Description Information or Supplementary information or identification Information or Other Forms of Information	Variable Field

Figure 4: CHDO Aggregation, MGDS SFDU Stream Data Structure

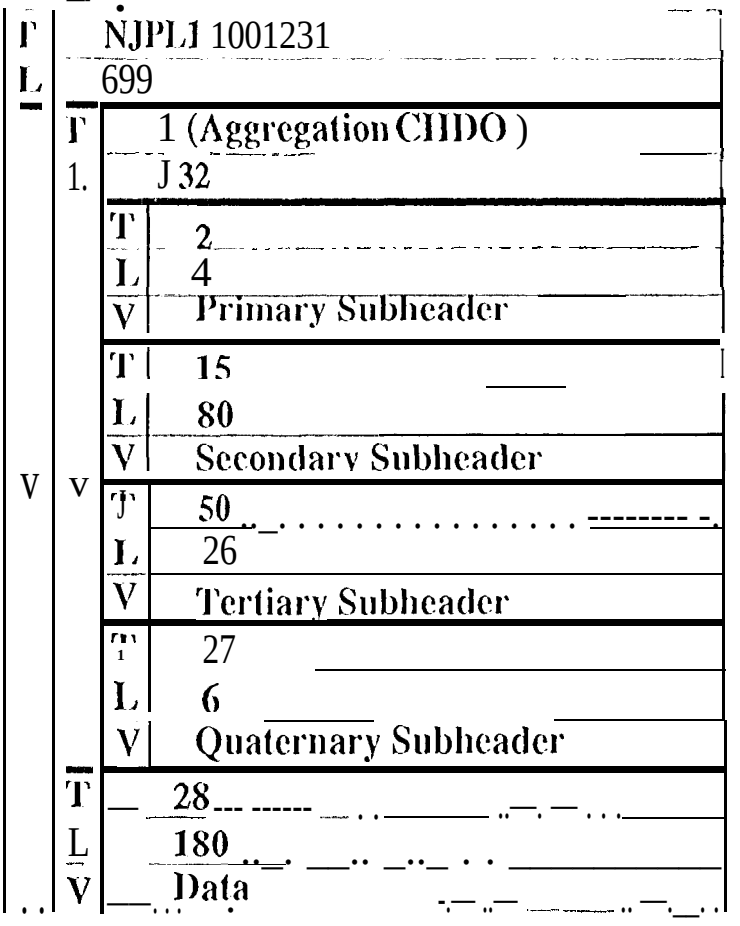

Figure 5: TDS Data Product

Primary Label	CCSD3ZS00001 TDSQDATA
K-HEADER Label	NJPL3KS0L009 TDSQUERY
Data Product Identification	OBJECT = QUERYNAME; ... END_OBJECT = QUERYNAME;
K-HEADER End Marker	CCSD3RE00000 TDSQUERY
Data & TDS Status Messages	NJPL...
End Marker Label	CCSD3RE00000 TDSQDATA

Figure 6: TDS Query Protocol

Primary, Label	CCSD3ZS00001
PVL SFDU Label	NJP131S01 .009
PVL Query Spee.	OBJECT = QUERYNAME; DESCRIPTION = " . . . " ; ... END_OBJECT= QUERYNAME;
PVL End	CCSD3RE00000
End Marker	CCSD3RE00000

Sample Query I'VI.

OBJECT = Mo. Query;
DESCRIPTION = 'Tot Query';
REQUESTER_NAME = Al Sacks;
MISSION. NAME = MO;
SPACECRAFT_NAME = Mo1;
TIME_TYPE = ERT;
START_TIME=91/352T20:09:;
END_TIME? = 91/352T21:09:;
GROUP = FRAME;
DATA_TYPE= sci_tes;
DSS_11)== ALL;
END_GROUP = FRAME;
END_OBJECT = Mo_Query;

Figure 7: Telemetry Output Tool